

```

1 Minu
2 -----
3     1.
4
5 Shankar
6 -----
7     1. Divide without if.
8     2. Top ranked words.
9
10 Ayush
11 -----
12     1. problem in MapEntry
13     2. dot product, how.
14
15
16 Saurab
17 -----
18     1. function composition problem.
19
20 =====
21 =====
22 Statements based language
23     * C, C++, Java, Python...
24     for(expr1;expr2;expr3){
25
26     }
27
28
29 Expressions based language
30     * Scala, Clojure
31
32     (defn -main ...)
33     {2 + 3 * () + ....}
34
35     (+ 2 3 ...)
36     Precedence, evaluates
37
38     Scala => def f1 = if(expr) arg1 else arg2
39     Clojure => (defn .. if cond arg1 arg2)
40
41 =====
42 Statements based langs =====> mid point <===== expr lgs
43
44 =====
45
46 partially defined function to total function =>
47
48 =====
49 (key, val) tuple
50
51 (into {} ...seq of tuple)
52
53 -----
54
55 V1 => {2 4, 5 1, 9 100} [0 0 4 0 0 1 0 0 0 100]
56 V2 => {5 3, 9 40, 100 4, 300 8}
57
58 5000 ngrams => dimensions 5000.
59
60 iterate 1 to 50000.
61
62 I dont want to go that far.
63
64 Accumulates: keys #{2, 5, 9} #{5, 9, 100, 300}

```

```

65
66 Union of two sets, converts to vec, sort.
67
68 [2, 9, 9, 100, 300] => iterate to these 5 values.
69
70 (get v1 2) * (get v2 2)
71 ...
72 reduce to +.
73
74 =====
75 Relevant Words Optimization
76 =====
77
78 1000 words
79
80 i_word => iterate to 1000 words
81
82 Inverted index:
83
84 player => {pla, lay, aye, yer, play, laye, ayer}
85
86 pla => [play, plays, plan]
87 lay => []
88
89 =====
90
91 [[play, plays, plan] [] []]
92 flatten, unique
93
94 100 words.
95
96 =====
97
98 function composition
99 -----
100 f(x) => x^....
101 g(x) =>
102
103 f(g(x)) =>
104 g(f(x)) =>
105
106 f(g(x)) != g(f(x))
107 -----
108 TF/IDF metric: term frequency (no. of appear) * inverse document frequency (no of word that
gram presence)
109 -----
110
111 System Evaluation
112 -----
113 Precision
114 Recall
115 FMeasure
116 -----
117 thesis: google (santa basnet theis)
118 -----
119
120
121
122
123
124
125
126
127

```

128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150